MSX

Talent

FLOPPY DISK DRIVE
USER'S MANUAL

DPF-550

(CPF-550/360)

Sr Ochoa.
MSX

MSX
FLOPPY DISK DRIVE
USER'S MANUAL

DPF-550

(CPF-550/360)

PREFACE

This operation manual is intended to be used as a reference guide whenever you wish to set-up, add to, or understand more about your MSX-FDD. This manual consist of the following three parts:

PART I. OPERATION MANUAL
PART II. MSX-DOS
PART III. MSX DISK-BASIC

PART I : OPERATION MANUAL is a description of how to use your FDD.

PART II : MSX-DOS is a description of the commands of MSX-DOS and its environment to help you use your diskette file effectively.

PART III : MSX DISK-BASIC is a description of the function and the format of the commands of DISK-BASIC.
It is on the basis of MSX ROM-BASIC.
For further descriptions in relation to DISK-BASIC, you should refer to the MSX-BASIC manual.

CONTENTS

Syntax Notation in Reference Sections

Wherever the format for a statement/command or a function is given, the following rules apply:

CAPS Items in capital letters must be input as shown.

< > Items in lowercase letters enclosed in angle brackets (< >) are to be suppied by the user.

[ ] Items in square brackets ([ ]) are optional.

... Items followed by an ellipsis (...) may be repeated any number of times (up to the length of the line).

{ } Braces indicate that the user has a choice between two or more entries. At least one of the entries enclosed in braces must be chosen unless the entries are also enclosed in square brackets.

| Vertical bars separate the choices within braces. At least one of the entries separated by bars must be chosen unless the entries are also enclosed in square brackets.

All punctuations except angle brackets and square brackets (i.e., commas, parentheses, semicolons, hyphens, equal signs) must be included where shown.

Arguments to functions are always enclosed in parentheses. In the formats given for the functions in this book, the arguments are abbreviated as follows:

X and Y Represent any numeric expressions.

I and J Represent integer expressions.

X$ and Y$ Represent string expressions.

PART I.
OPERATION MANUAL

Before you start to use this equipment you should read this reference manual to fully understand and utilize it's outstanding features.

This reference manual explains the proper usage of the minifloppy disk driver that can be connected to the MSX computer of our company and also a few points of caution in it's usage.

CHAPTER ONE

INTRODUCTION TO CPF-555/360 SERIES

The CPF-555/360 series is secondary memory device developed to support the MSX-computer capable of connecting two slim-type mini floppy disk drives at a time. Thise series can also be connected to any other MSX personal computer and does not need any supplementary software for use of the MSX DISK-BASIC.

MSX-DOS is used with the 64KB model and because of it's data compatability with the IBM PC, files any be used commonly moving among diskettes. MSX-DOS is also function compatible with the CP/M, which means that you can utilize existing software of the CP/M.

Data memory capacity is 250KByte or 500KByte (unformatted) for CPF-555 series and 500KByte or 1MByte (unformatted) for CPF-360 series. You can access the library of software available on disk or write the program and data. You can use utility programs in MSX-DOS, DISK-BASIC and CP/M.

1-1. Packing list

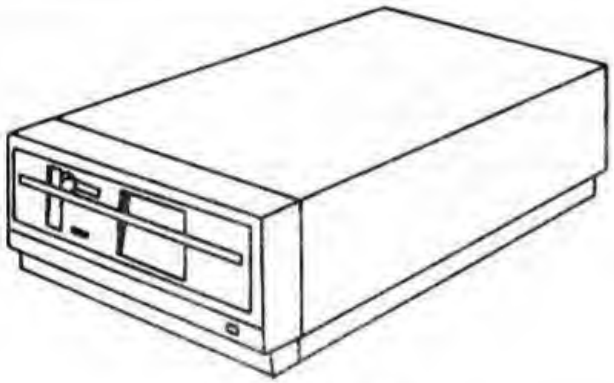
Your MSX-FDD is packed in poly-foam carefully. Save it and use when transporting the disk drive.

The followings are contained in the carton.

1. CPF-555 or 360 series MSX-FDD
2. This manual (User's manual)
3. CP/M operating system manual (CP/M option)
4. MSX-DOS system disk
5. CP/M OS disk (CP/M option)

1-2. CPF-555 series.

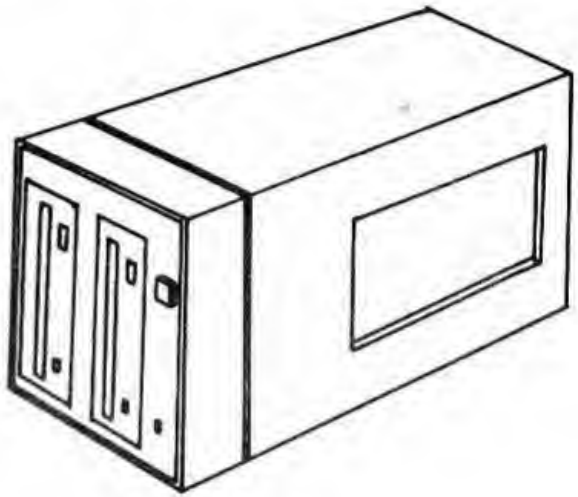
The CPF-555 series are all MSX-FDD and use the 5.25 inch flexible disk. Their specifications are shown at below figure.



CPF-555 series

1-3. CPF-360 series

The CPF-360 series are all MSX-FDD and use the 3.5 inch micro-flexible disk. Their specifications are shown at below figure.



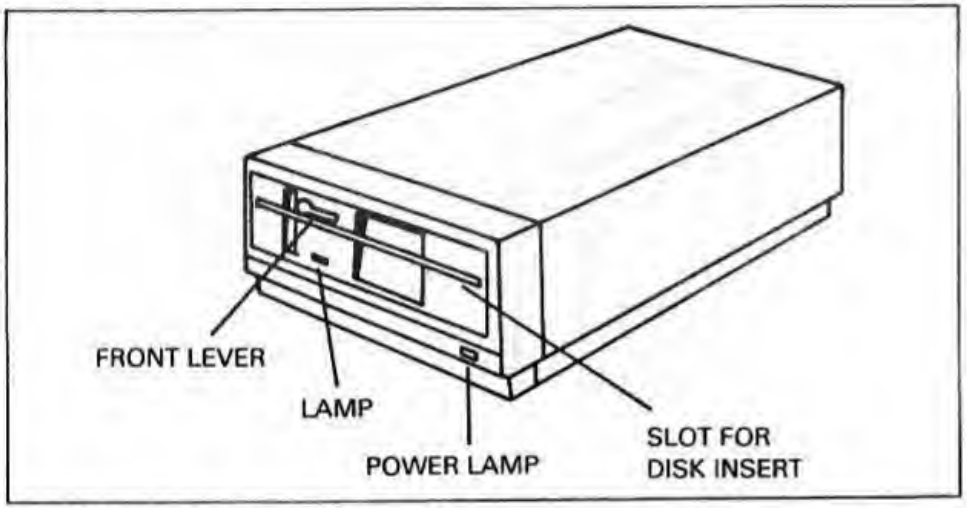
CPF-360 series

CHAPTER TWO

CONFIGURATIONS OF THE CPF-555 SERIES

2.1. Names and Functions of each part

2.1.1 Front Side



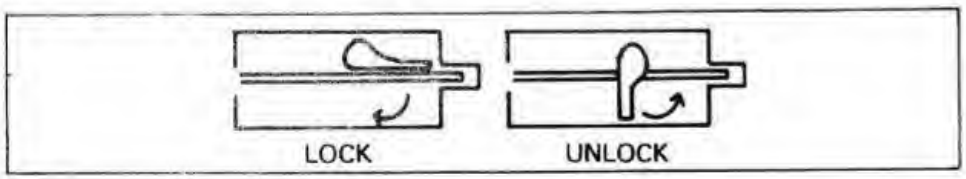
(1) Power Switch and Power Lamp

When the switch is 'ON', power is being supplied and when so, the red power lamp turns on.
When the switch is turned to 'OFF', the power is disconnected and the power lamp is turned off.

(2) Front Lever

When first packed, there is a disk-like paper slipped into the disk drive to protect the disk head. This disk-like paper is not actually usable, but keep it somewhere safe and use it to protect the disk head in case of transport.

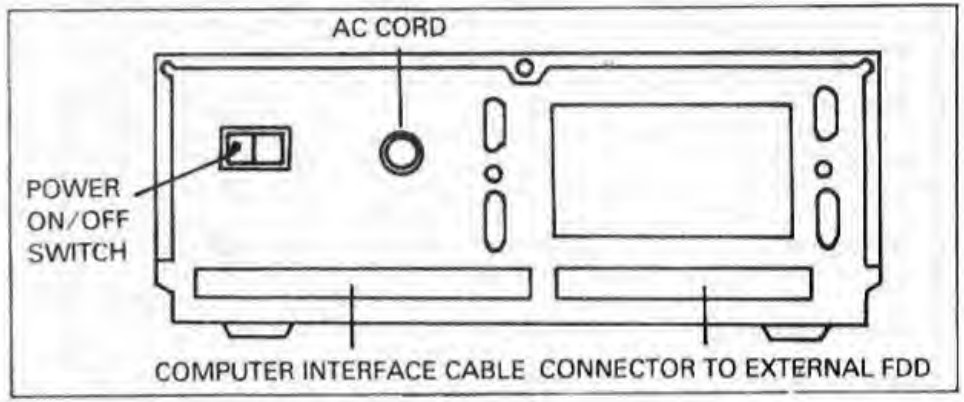
When a diskette is slipped in, the front lever must be turned to the right to fix the diskette. To take the diskette out, turn the lever to the left and pull the diskette out.



(3) Drive In use Lamp

When one disk drive is used, the computer assumes that only the A-DISK is present and in case of two drives it assumes that both A-DISK and B-DISK are present. The drive connected to the computer is the A-DISK and the other drive is the B-DISK. When the software selects one of the drives, the lamp of the drive is turned on.

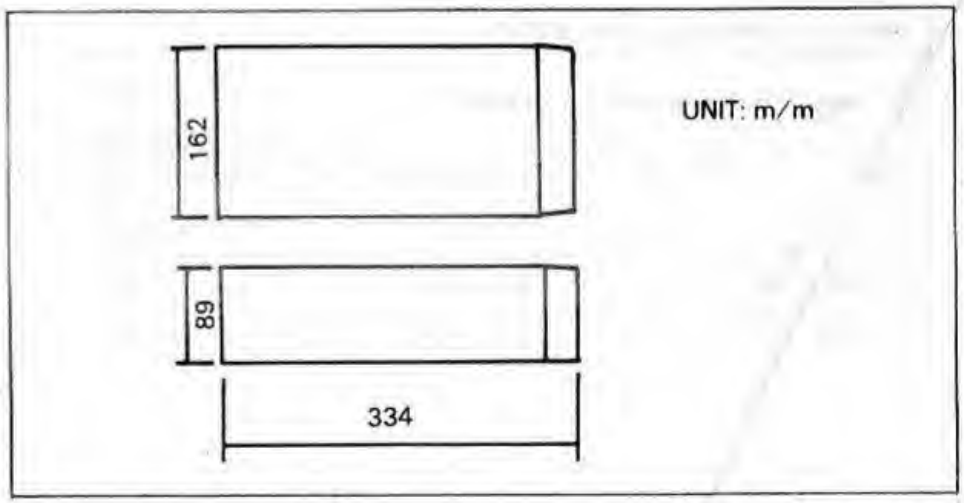
2.1.2 Rear Side



Connector to the External FDD

This is the part where you connect the connecting apparatus which was packed with the disk drive. The connection is possible in only one direction, so if it doesn't connect easily, do not try to jam it in but check if it is in the right direction.

2.2 Outer Measurements



2.3 Specifications

2.3.1 Memory capacity and Disk Drive Characteristics (for 1 drive)

	CPF-555S	CPF-555SC	CPF-555D	CPF-555DC
Capacity (unformatted)	250KB	250KB	500KB	500KB
Recording format	MFM	MFM	MFM	MFM
Track per disk	40	40	40×2	40×2
sector per disk	9	9,(17)	9	9,(17)
Sector size	512byte	512byte (256byte)	512byte	512byte (256 byte)
Operating system	MSX-DOS	MSX-DOS (CP/M)	MSX-DOS	MSX-DOS (CP/M)
Disk (5.25inch)	1S/DD	1S/DD	2S/DD	2S/DD

( ): CP/M option
1S/DD: 1side double density
2S/DD: 2side double density

2.3.2 Power Source

Input power	AC 220V ± 10%, 50Hz
Power consumption	20W (MAX)

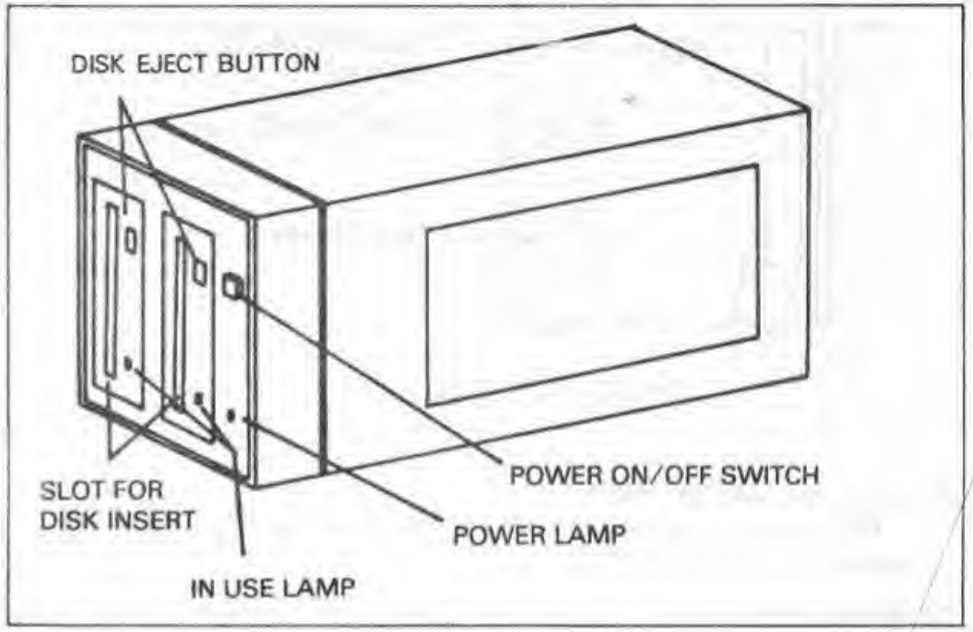
2.3.3. Usage Environmental Conditions

State	Temperature	Relative Humidity
In use	10 ~30°C	20 ~80%
Transporting	–40 ~ 60°C	5 ~95%
Storage	–20 ~ 40°C	5 ~95%

CONFIGURATIONS OF THE CPF-360 SERIES

2.4 Names and Functions of each part

2.4.1 Front side



(1) Power Switch and power lamp

When the switch is "ON", the power is being supplied and the red power lamp turns "ON" when the switch is "OFF", the power lamp turns off.

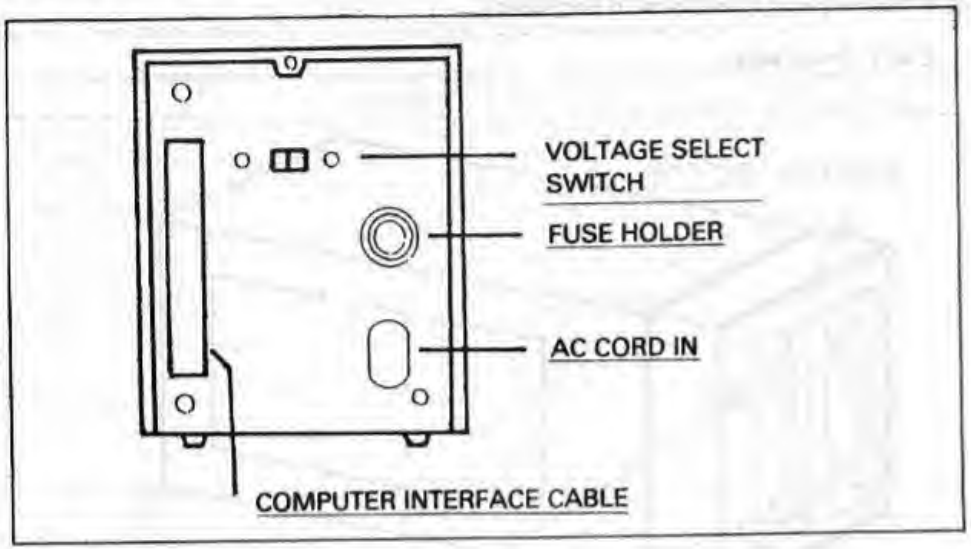
(2) Disk eject button

To eject your disk press this button and the disk pops out.

(3) Drive in use lamp

When one disk drive is used, the computer assumes that only the A-DISK is present and in case of two drives, it assumes that both A-DISK and B-DISK are present. When the software selects one of the drives, the lamp of the drive turns on.

2.4.2. Rear side



(1) Computer interface cable

This cable is to be connected to the rear slot of your MSX-computer. Connect it in correct direction.

(2) Voltage select switch

If your FDD is not dual voltage, this switch does not exist.

2.5 Specifications

2.5.1 Memory capacity and Disk drive characteristics

	CPF-350	CPF-350C	CPF-360	CPF-360C
Capacity (unformatted)	500KB	500KB	IMB	IMB
Recording format	MFM	MFM	MFM	MFM
Track per disk	80	80	80	80
Sector per disk	9	9,(17)	9	9,(17)
Sector size	512byte	512byte (256byte)	512byte (256byte)	512byte (256 byte)
Operating system	MSX-DOS	MSX-DOS (CP/M)	MSX-DOS	MSX-DOS (CP/M)
Disk (3.5inch)	1S 135TPI	1S 135TPI	1S 135TPI	1S 135TPI

( ); CP/M option
1S: 1side disk

2.5.2 Power Source

AC INPUT POWER	220V AC ± 10%, 50Hz
Power consumption	15W (max)

2.5.3. Usage Environmental Conditions

State	Temperature	Relative Humidity
IN USE	10 ~ 30°C	20 ~ 80% RH
Transporting	–40 ~ 60°C	5 ~ 95% RH
Storage	–20 ~ 40°C	5 ~ 95% RH

2.6 Precautions

(1) When switch is turned OFF, wait at least 10 seconds before turning it on again.

(2) Always use specified power supply.

(3) Install where ventilation is easy and at least 10cm away from the wall.

(4) When in transport, be sure to put in the head protect paper.

(5) Dirt or cigarrette smoke may damage head and diskette surface, so keep the surrounding clean.

(6) Do not put heavy objects on the device.

(7) Keep chemical substance and water (moist) away from the device.

(8) When cleaning the case do not let water leak into the device. Do not use volatile detergents when cleaning.

(9) Keep magnetic objects away.

CHAPTER THREE

SETTING UP

Follow the procedure outlined below to connect the disk drive to your computer.

(1) Confirm the equipment

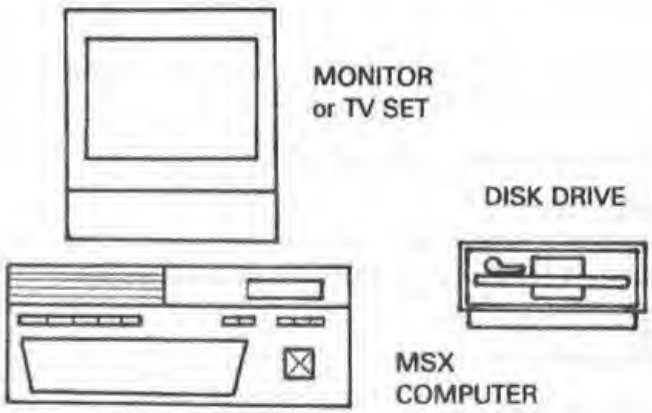
Unpack and check if the following are all there.

a) Disk drive
b) User's manual
c) CP/M operating system manual (CP/M option)
d) MSX-DOS disk
e) CP/M OS disk (CP/M option)

(2) Prepatory equipment

The following is necessary to use the disk drive.

a) MSX computer (RAM 64KB or over)
b) Monitor or TV set
c) Disk (MSX-DOS or CP/M OS)
d) Power source (AC)

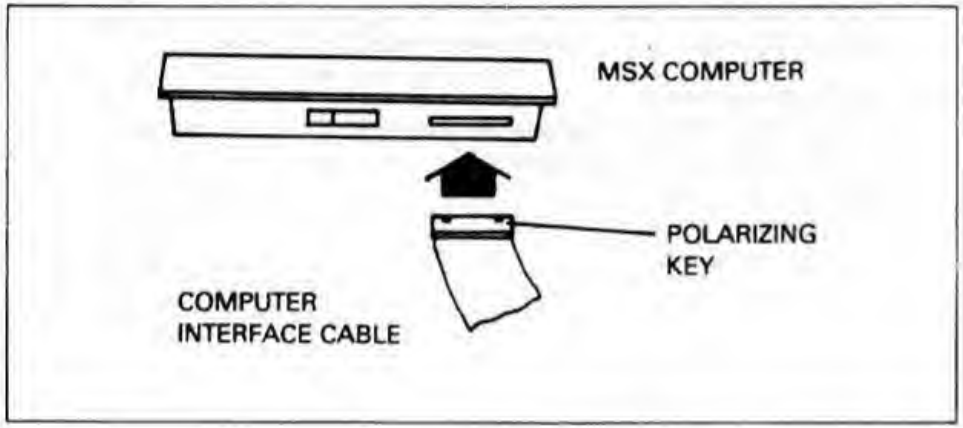


(3) Main frame connection

Connect the main frame (computer) and monitor (or TV set). On how to connect, refer to your computer manual.

(4) Connection with the MSX computer

Place the disk drive 10 to 30cm to the right side of the computer. Position the disk drive cable so that the polarizing key of the connector is on top. Connect the computer interface cable to the expansion slot at the rear of the computer.



(5) Close the latch

Close the latch of the expansion slot carefully.

(6) Confirm the connection

Check if the connections are correct.
Caution must be taken, for wrong connections may cause defects.

(7) Connect the AC power cord.

Connect the AC power source.

(8) Reconfirmation Reconfirmation

Check again if the steps (1) through (7) have been correctly carried out.

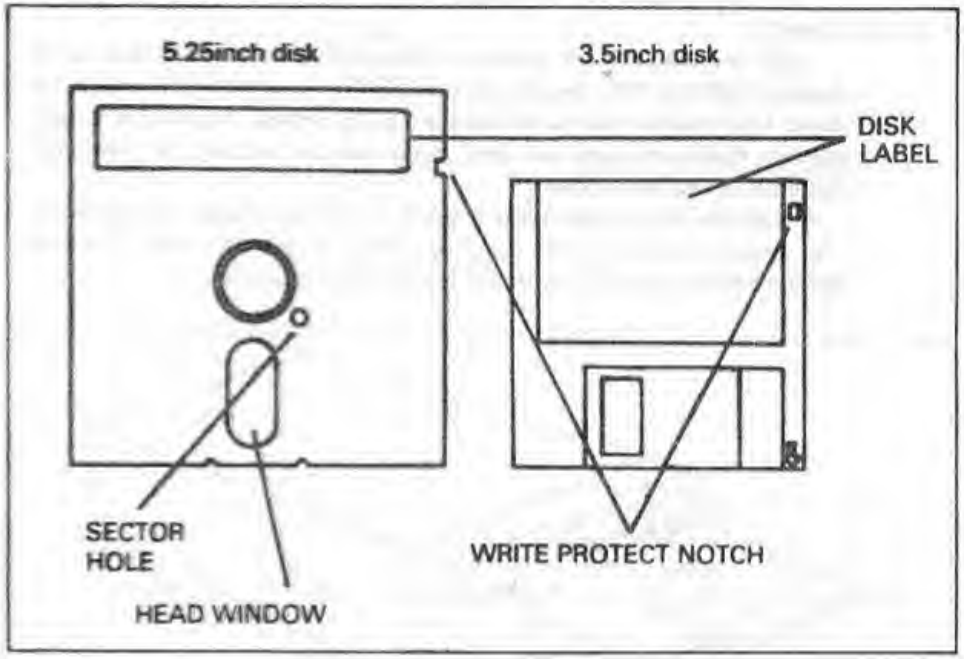
(9) Power switch ON

Switch your disk drive on before the computer. That is, turning on of the switches should be done in following sequence.

a) Monitor power on
b) Disk drive power on

(10) Insert disk

Insert your disk in to the disk drive with the disk label facing upwards and the head window towards the slot on the disk drive.
If your system is CPF-555 series, turn the front lever to the right side.



(11) Power on your computer

When your computer turns "ON", the disk drive starts booting CP/M OS or MSX-DOS.

(12) Power OFF procedures

a. Remove the disk
b. Monitor power off
c. FDD power off
d. Computer power off
e. Pull out the power plug from power source.

CHAPTER FOUR

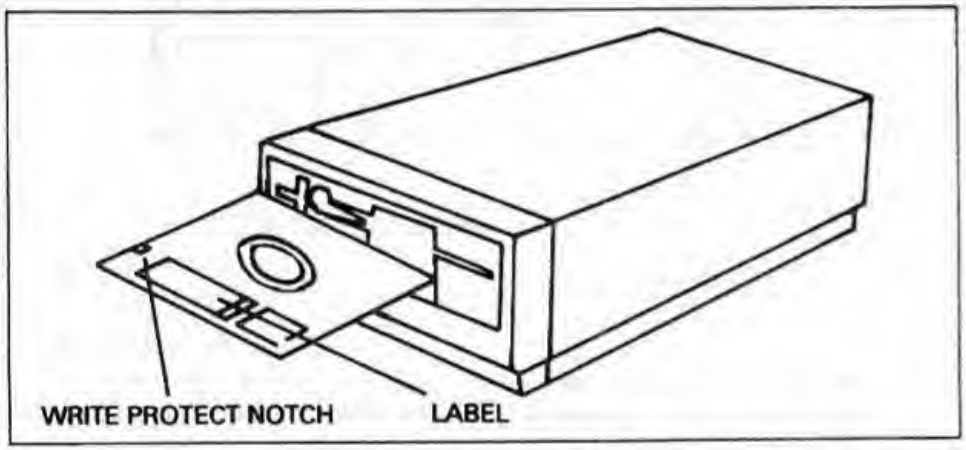
HOW TO USE

4.1 CPF-555 Series

4.1.1 Diskette

Since the disk drive is double density the diskette must also be of double density. This means that a diskette marked "2D" must be used. The diskette size is of standard 5.25 inches. The use of single density diskettes may not give good results, so use of standard diskettes is recommended.

A diskette has a side with a label on it and a side that doesn't. The direction in which the diskette is slipped into the drive is as the diagram below; the side with the label facing upwards.



To read but protect the diskette from being written on, simply cover the Write Protect Notch.

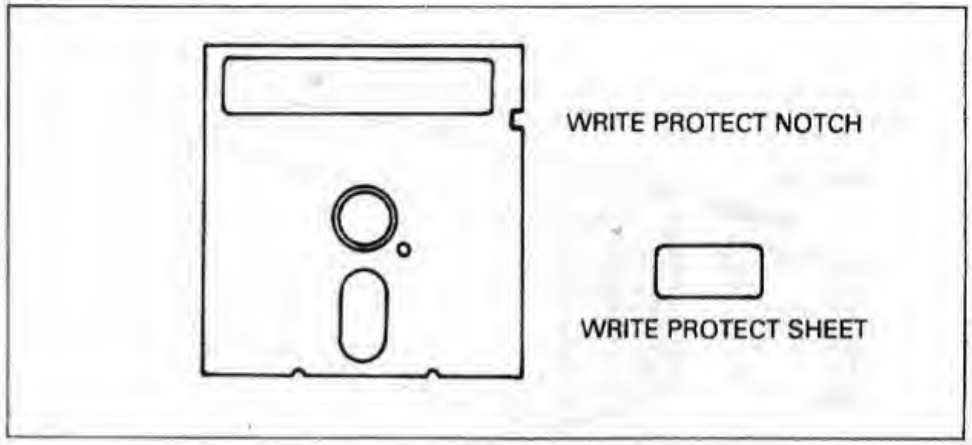
4.1.2 Standard diskette

Your disk drive needs following diskette:

CPF-555S/CPF-555SC : 5.25 inch single side double density 48 TPI or over

CPF-555D/CPF-555DC : 5.25 inch double side double density 48 TPI or over

4.1.3 Write Protect your disk

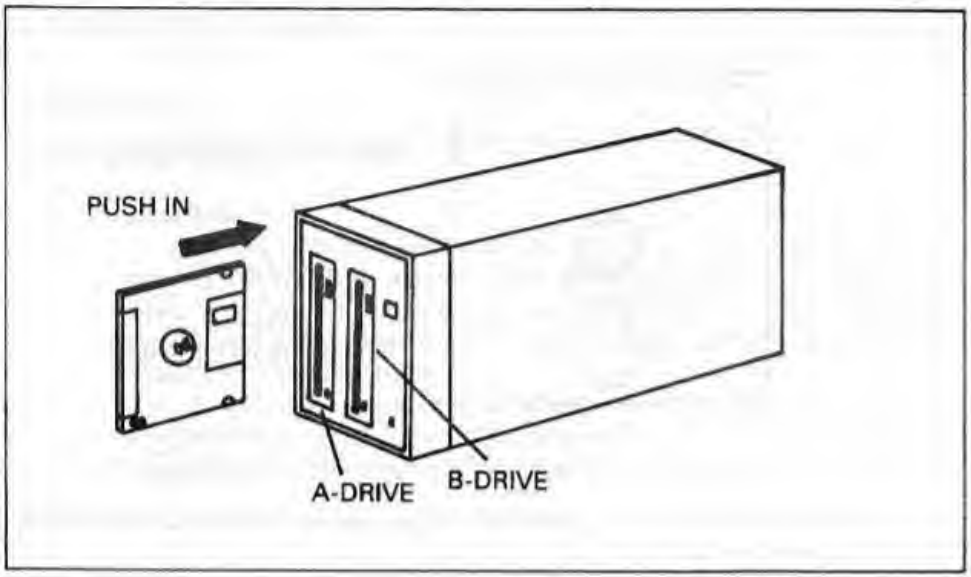


When the notch is sealed with write protect sheet, the disk cannot be written on.

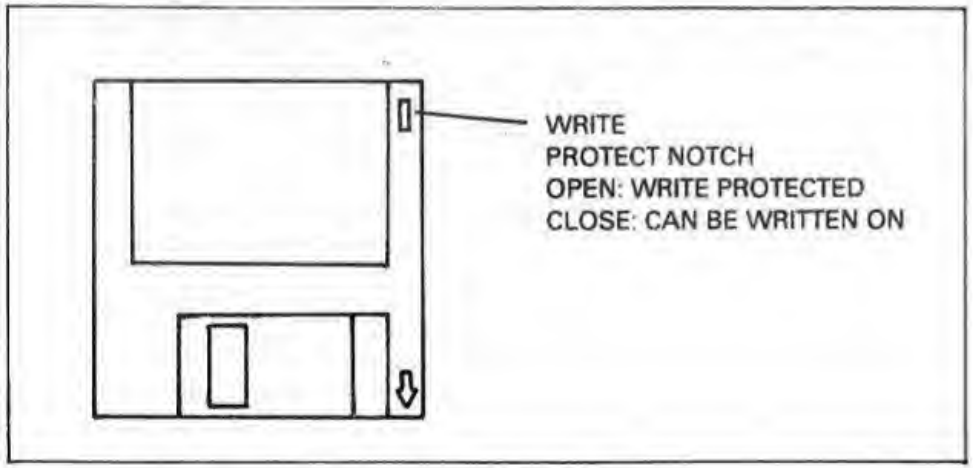
4.2 CPF-360 Series

4.2.1 Diskette

Since the disk drive is single side 80 track, the diskette must be of single side 135 TPI. The diskette size is of standard 3.5 inches. The use of 67.5 TPI diskettes may not give good results, so use of standard diskette is recommended. The direction in which the diskette is slipped into the drive is as the diagram below; the side with the label facing right sides.



4.2.2. Write protect notch



CARING NOTICE OF DISKETTE

The following rules apply to diskettes

1. Do not touch, soil or scratch the recording surface (i.e., any exposed area on the diskette especially the head window). It is never necessary to turn the diskette by hand.
2. Do not bend or fold
3. Keep diskettes away from magnetic field, (motors, TV sets, telephones, etc). Strong magnetic fields can distort recorded data.
4. Do not write on the disk label with a lead pencil or ball point pen. Use a felt tip pen instead.
5. When a diskette is not in use, put it in the envelope and file it away in a safe place. Replace storage envelopes when worn, cracked or distorted.
6. A temperature range for diskette is 4~50°C
7. Do not place diskettes in direct sunlight

CHAPTER FIVE

OPTIONS

5.1 MSX-DOS UTILITIES

High level languages supported by MSX-DOS are FORTRAN, COBOL, etc. of the Microsoft company. MSX-DOS also supports useful utility packages of MULTIPLAN and Macro Assembler, Linker, Sort and other utilities.

5.2 CP/M

You can use CP/M with MSX-computer and the CPF-555/360 option series. CP/M OS (operating system) is stored in disk hence if you have a CP/M OS disk for MSX-computer, you can use CP/M utilities. If you have only MSX-DOS FDD and to be able to use CP/M bring your disk drive to your agency or distributor for hardware modification.

CHAPTER SIX

SHORT CHECK OF OPERATION

Defect	Check Point	Solution
1. The drive does not work.	1. Check if the power lamp is on.	1. Plug the cord. 2. Turn the power switch on. 3. Supply power.
	2. Is the drive motor running?	1. Connect the cables correctly. 2. Check if the computer is normal.
2. READ and WRITE error.	1. Is the diskette in the right direction. 2. Is the diskette in the right position. 3. Is the front lever fixed 4. Has the diskette been damaged.	1. Put, it in the right direction. 1. Take it out and put it in the right position. 1. Turn the lever to the right to fix it. 1. Use a new diskette.
3. WRITE error	1. Is the write protect notch closed.	1. Open the notch.

** For malfunctions other than the above, contact your agency or distributor for After Service repair.

CHAPTER SEVEN

SYSTEM DISK BOOTING

7.1 MSX-DOS or DISK-BASIC booting

1 Set up your disk drive. [refer chapter 3 (1) ~ (9)]
2. Insert the disk labeled "MSX-DOS SYSTEM DISK" into the disk drive.
3 Switch "ON" the computer. The computer will boot "MSXDOS. SYS" and "COMMAND. COM" from the disk drive. Then MSX-DOS system prompt "A>" will appear on the monitor screen.

```
MSX-DOS version 1.00
Copyright 1984 by Microsoft

Command version 1.01

Current date is Sun 1984-01-01
Enter new date:—
```

4. If you want to run DISK-BASIC or MSX-BASIC, then type [B][A][S][I][C] or [b][a][s][i][c] and press [RETURN] key.
5. The monitor screen by DISK-BASIC will be shown as.

```
MSX BASIC version 1.0
Copyright 1983 by Microsoft
22354 Bytes free
DISK BASIC version 1.0
OK
■
```


[NOTE]

If you have the CP/M option disk drive, the computer displays the message:
"loading CP/M system ....."
then restarts the booting of "MSX-DOS SYSTEM".

7.2 CP/M Booting (CP/M option series)

1. Set up your disk drive. [refer chapter 3 (1) ~ (9)]
2. Insert the disk labeled "CP/M OS DISK" into the disk drive.
3. Switch "ON" the computer. The computer will boot the "CP/M OS" from the disk drive. Then CP/M OS prompt "A>" will appear on the monitor screen.

```
MSX CP/M-80 Revision 2.24
for CPF-555/360 (release 1.0)
Copyright (c) by Digital Research
A>—
```

4. In this mode you can use only CP/M, cannot use either DISK-BASIC or MSX-DOS. If you want to use MSX-DOS or DISK-BASIC, reboot your FDD with MSX-DOS diskette.

PART II.
MSX-DOS

II-1. MSX-DOS

MSX-DOS is a disk operating system for MSX computers. The system with its compatibility to other versions of MSX-DOS will surely provide you a comfortable environment around. All Microsoft languages (BASIC Interpreter, BAISC Compiler, FORTRAN, COBOL, Pascal) will be available under MSX-DOS. Users of MSX-DOS are assured that their operating system will be the first that Microsoft will support when any new products or major releases are announced.

II—2. MSX-DOS User's Guide

II-2.1 System Requirements

The MSX-DOS operating system requires a MSX microcomputer system with 64k bytes of memory (RAM) and at least one disk drive.

The MSX-DOS disk contains the following files:

File Name	Function of File
COMMAND. COM	MSX-DOS command processor
MSXDOS. SYS	MSX-DOS operating system

II-2.2. Getting Started

Once MSX-DOS has been loaded, the system searches the MSX-DOS disk for the COMMAND. COM file and loads it into memory. The COMMAND. COM file is a program that processes the commands you enter and then runs the appropriate programs. It is also called the command processor.

When the command processor is loaded, you will see the following display on your screen (the underscore represents the cursor):

```
MSX-DOS Version 1.00
Copyright 1984 by Microsoft

Command version 1.01

Current date is Sun 1984-01-01
Enter new date:-
```


Any date is acceptable in answer to the new date prompt as long as it follows the above format. Separators between the numbers can be hyphens (-) or slashes (/).

After you have answered the new time prompt, the MSX-DOS

```
A>_
```

will be displayed.

It tells you that MSX-DOS is ready to accept commands. If you have inserted the MSX-DOS disk into a drive other than A, the command processor prompt will reflect that drive (for example, B>). However, usually you will load MSX-DOS in drive A.

The A in the previous prompt represents the default disk drive. This means that MSX-DOS will search only the disk in drive A for any filenames you may enter and will write files to that disk unless you specify a different drive. You can ask MSX-DOS to search the disk in drive B by changing the diver designation or by specifying B: in a command. To change the disk drive designation, enter the new drive letter followed by a colon. For example:

```
A>      (MSX-DOS prompt)
A> B:   (you have typed B: in response to the prompt)
B>      (system responds with B    and drive B
         is now the default drive)
```

The system prompt B> will appear and MSX - DOS will search only the disk in drive B until you specify a different default drive.

If you have only one disk drive attached to your computer, turn to 3.1.14 Instructions for Users with Single-Drive Systems, for important information.

A filename can be from 1 to 8 characters long. The filename extension can be three or fewer characters. You can type any filename in small or capital letters and MSX-DOS will translate these letters into uppercase characters.

In addition to the filename and the filename extension the name of your file may include a drive designation. A drive designation tells MSX-DOS to look on the disk in the designated drive to find the filename typed.

The following characters are allowed for file names and their extensions.

A-Z 0-9 $ & #
% ' ( ) — @
Y ^ { } ~ \ !

(A backslash instead of Yen sign in international versions.)

The term file specification (or filespec) will be used in this book to indicate the following filename format:

[<drive designation:>] <filename> [<.filename extension>]

II-2.3 Wild Cards

Two special characters (called wild cards) can be used in filenames and extensions: the asterisk (*) and the question mark (?). These special characters give you greater flexibility when using filenames in MSX-DOS commands.

° The ? Wild Card

A question mark (?) in a filename or filename extension indicates that any character can occupy that position. For example, the MSX-DOS command

DIR TEST?RUN. COM

will list all directory entries on the default drive that have 8 characters, beginning with TEST, have any next character, and with the letters RUN, and have a filename extension of. COM.

° The * Wild Card

An asterisk (*) in a filename or filename extension indicates that any character can occupy that position or any of the remaining positions in the filename or extension.
For example:

DIR TEST *. COM.

will list all directory entries on the default drive with filenames that begin with the characters TEST and have an extension of. COM.

The will card designation *.* refers to all files on the disk. Note that this can be very bowerful and destructive when used in MSX-DOS commands. For example, the command DEL *.* deletes all files on the default drive, regardless of filename or extension.

II-2.4 Illegal File Names

MSX-DOS treats some device names specially, and certain 3-letter names are reserved for the names of these devices. These 3-letter names cannot be used as filenames or extensions. You must not name your files with any of the following:

AUX	Used when referring to input from or output to an auxiliary device (such as a printer or disk drive).
CON	Used when referring to keyboard input or to output to the terminal console (screen).
LST or PRN	Used when referring to the printer device.
NUL	Used when you do not want to create a particular file, but the command requires an input or output filename.

Even if you add device designations or filename extensions to these filenames, they remain associated with the devices listed above. For example, A:CON.XXX still refers to the console and is not the name of a disk file.

II-2.5 Directories

The directory also contains information on the size of the files, their locations on the disk, and the dates that they were created and updated.

II-2.6 Types of MSX-DOS Commands

There are two types of MSX-DOS commands:

Internal commands

External commands

Internal commands are the simplest, most commonly used commands. You cannot see these commands when you do a directory listing on your MSX-DOS disk; they are part of the command processor. When you type these commands, they execute immediately. The following internal commands are described in 3.2.

BASIC	DIR	REM
COPY	FORMAT	REN (RENAME)
DATE	MODE	TIME
DEL (ERASE)	PAUSE	TYPE

External commands reside on disks as program files. They must be read from disk before they can execute. If the disk containing the command is not in the drive, MSX-DOS will not be able to find and execute the command.

Any filename with a filename extension of .COM or .BAT is considered an external command. For example, programs such as FILCON.COM and COMP.COM are external commands. Because all external commands reside on disk, you can create commands and add them to the system. Programs that you create with most languages (including assembly language) will be .COM (executable) files.

When you enter an external command, do not include its filename extension.

II-2.7 Command Options

Options can be included in your MSX-DOS commands to specify additional information to the system. If you do not include some options, MSX-DOS provides a default value.

The following is the format of all MSX-DOS commands:

Command [options...]

where:

switches — Switches are options that control MSX-DOS commands. They are preceded by a slash (for example, /P).

arguments — Provide more information to MSX-DOS commands. You usually choose between arguments; for example, ON or OFF.

filespec — Refers to an optional drive designation, a filename, and an optional three letter filename extension in the following format:

[<d:>] <filename> [<.ext>]

d: — Refers to a disk drive designation.

filename — Refers to any valid name for a disk file, including an optional filename extension.
The filename option does not refer to a device or to a disk drive designation.

.ext — Refers to an optional filename extension consisting of a period and 1-3 characters.
When used, filename extensions immediately follow filenames.

II-2.8 Information Common to All MSX-DOS Commands

The following information applies to all MSX-DOS commands:

- Commands are usually followed by one or more options.
- Commands and options may be entered in uppercase or lowercase, or a combination of keys.
- Commands and options must be separated by delimiters. Because they are easiest, you will usually use the space and comma as delimiters. For example:

  ```
  DEL MYFILE. OLD NEWFILE. TXT
  RENAME, THISFILE THATFILE
  ```

  You can also use the semicolon (;), the equal sign (=), or the tab key as delimiters in MSX-DOS commands.

- Do not separate a file specification with delimiters, since the colon and the period already serve as delimiters.
- When instructions say "Strike a key when ready", you can press any key except < CONTROL-C >.
- You must include the filename extension when referring to a file that already has a filename extension.
- You can abort commands when they are running by pressing < CONTROL-C >.
- Commands take effect only after you have pressed the <RETURN> KEY.
- Wild cards (global filename characters) and device names (for example, PRN or CON) are not allowed in the names of any commands.
- When commands produce a large amount of output on the screen, the display will automatically scroll to the next screen. You can press <CONTROL-S> to suspend the display. Press any key to resume the display on the screen.
- MSX-DOS editing and function keys can be used when entering commands. Refer to 3.1.13 MSX-DOS Editing and Function Keys, for a complete description of these keys.

- The prompt from the command processor is the default drive designation plus a right angle bracket (>); for example, A >.
- Disk drives will be referred to as source drives and destination drives. A source drive is the drive you will be transferring information to.

II-2.9 Batch Processing

With MSX-DOS, you can put the command sequence into a special file called a batch file, and execute the entire sequence simply by typing the name of the batch file. "Batches" of your commands in such files are processed as if they were typed at a terminal. Each batch file must be named with the .BAT extension, and is executed by typing the filename without its extension.

Two MSX-DOS commands are available for use expressly in batch files: REM and PAUSE. REM permits you to include remarks and comments in your batch files without these remarks being executed as commands. PAUSE prompts you with an optional message and permits you to either continue or abort the batch process at a given point.

The following list contains information that you should read before you execute a batch process with MSX-DOS:

- Do not enter the filename BATCH (unless the name of the file you want to execute is BATCH.BAT).
- Only the filename should be entered to execute the batch file. Do not enter the filename extension.
- The commands in the file named < filename > .BAT are executed.
- If you press < CONTROL-C > while in batch mode, this prompt appears:

  Terminate batch job (Y/N)?

  If you press Y, the remainder of the commands in the batch file are ignored and the system prompt appears.

  If you press N, only the current command ends and batch processing continues with the next command in the file.

- If you remove the disk containing a batch file being executed, MSX-DOS prompts you to insert it again before the next command can be read.
- The last command in a batch file may be the name of another batch file. This allows you to call one batch file from another when the first is finished.

II-2.10 The AUTOEXEC.BAT File

When you start MSX-DOS, the command processor searches the MSX-DOS disk for a file named AUTOEXEC.BAT. The AUTOEXEC.BAT file is a batch file that is automatically executed each time you start the system.

If MSX-DOS finds the AUTOEXEC.BAT file, the file is immediately executed by the command processor and the date prompts are bypassed.

If MSX-DOX does not find an AUTOEXEC.BAT file when you first load the MSX-DOS disk, then the date and time prompts will be issued.

II-2.11 How to Create a Batch File

If, for example, you want to automatically load BASIC and run a program called MENU each time you start MSX-DOS, you sould create an AUTOEXEC.BAT file as follows:

1. Type:

   COPY CON: AUTOEXEC.BAT

   This statement tells MSX-DOS to copy the information from the console (keyboard) into the AUTOEXEC.BAT file. Note that the AUTOEXEC.BAT file must be created in the root directory of your MSX-DOS disk.

2. Now type:

   BASIC MENU

   This statement goes into the AUTOEXEC.BAT file.
   It tells MSX-DOS to load BASIC run the MENU program whenever MSX-DOS is started.

3. Press the <CONTROL-Z> key; then press the <RETURN> key to put the command BASIC MENU in the AUTOEXEC.BAT file.

4. The MENU program will now run automatically whenever you start MSX-DOS.

To run your own BASIC program, enter the name of your program in place of MENU in the second line of the example. You can enter any MSX-DOS command or series of commands in the AUTOEXEC.BAT file.

NOTE

Remember that if you use an AUTOEXEC.BAT file, MSX-DOS will not prompt you for a current date unless you include the DATE command in the AUTOEXEC.BAT file. It is strongly recommended that you include this command in your AUTOEXEC.BAT file, since MSX-DOS uses this information to keep your directory current.

II-2.12 Replaceable Parameters in .BAT Files.

There may be times when you want to create an application program and run it with different sets of data. These data may be stored in various MSX-DOS files.

When used in MSX-DOS commands, a parameter is an option that you define. With MSX-DOS, you can create a batch (.BAT) file with dummy (replaceable) parameters. These parameters, named %0-%9, can be replaced by values supplied when the batch file executes.

For example, when you type the command line COPY CON MYFILE.BAT, the next lines you type are copied from the console to a file named MYFILE.BAT on the default drive:

```
A > COPY CON MYFILE.BAT
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

Now, press <CONTROL-Z> and then press <RETURN>. MSX-DOS responds with this message:

```
1 File (s) copied
A > _
```

The file MYFILE.BAT, which consists of three commands, now resides on the disk in the default drive.

The dummy parameters %1 and %2 are replaced sequentially by the parameters you supply when you execute the file. The dummy parameter %0 is always replaced by the drive designator, if specified, and the filename of the batch file (for example, MYFILE).

NOTES:

1. Up to 10 dummy parameters (%0-%9) can be specified.
2. If you use the percent sign as part of a filename within a batch file, you must type it twice. For example, to specify the file ABC%-.COM, you must type it as ABC%%.COM in the batch file.

To execute the batch file MYFILE.BAT and to specify the parameters that will replace the dummy parameters, you must enter the batch filename (without its extension) followed by the parameters you want MSX-DOS to substitute for %1, %2, etc.

Remember that the file MYFILE.BAT consists of 3 lines:

```
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

To execute the MYFILE batch process, type:

```
MYFILE A:PROG1 B:PROG2
```

MYFILE is substituted for %0, A:PROG1 for %1, and B:PROG2 for %2.

The result is the same as if you had typed each of the commands in MYFILE with their parameters, as follows:

```
COPY A:PROG1.MAC B:PROG2.MAC
TYPE B:PROG2.PRN
TYPE MYFILE.BAT
```

The following table illustrates how MSX-DOS replaces each of the above parameters:

BATCH FILENAME	PARAMETER1 (%0) (MYFILE)	PARAMETER2 (%1) (PROG1)	PARAMETER3 (%2) (PROG2)
MYFILE	MYFILE.BAT	PROG1.MAC	PROG2.MAC PROG2.PRN

Remember that the dummy parameter %0 is always replaced by the drive designator (if specified) and the filename of the batch file.

II-2.13 MSX-DOS Editing and Function Keys

Special MSX-DOS Editing Keys

Control Character Functions

II-2.13.1 Special MSX-DOS Editing Keys

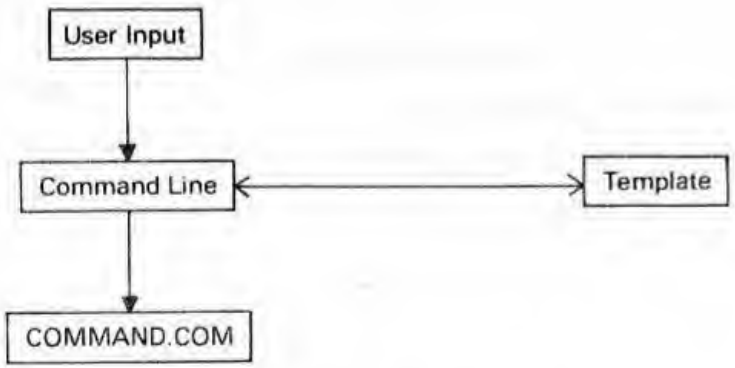
The special editing keys deserve particular emphasis because they depart from the way in which most operating systems handle command input. You do not have to type the same sequences of keys repeatedly, because the last command line is automatically placed in a special storage area called the template.

By using the template and the special editing keys, you can take advantage of the following MSX-DOS features:

- A command line can be instantly repeated by pressing two keys.
- If you make a mistake in the command line, you can edit it and retry without having to tetype the entire command line.
- A command line that is similar to a preceding command line can be edited and executed with a minimum of typing by pressing special editing keys.

When you type a line to the system call 0AH (buffered line input) and press the RETURN key, the line is returned to the caller of the system call. This line is copied to the new template. You can now recall the last line or modify it with MSX-DOS special editing keys.

The relationship between the command line and the template is shown in the next figure.



Command Line and Template

NAME	KEY	FUNCTION
COPY1	RIGHT ARROW ^Y (*)	Copies one character from the template to the new line.
COPYUP	SELECT ^X	Copies all characters from the template to the new line, up to the character specified.
COPYALL	DOWN ARROW ^	Copies all remaining characters in the template to the new line.
SKIP1	DEL	Skips over (does not copy) a character in the template.
SKIPUP	CLS ^L	Skips over (does not copy) the characters in the template, up to the character specified.
VOID	UP ARROW ESCAPE ^^ ^U ^[	Voids the current input. Leaves the template unchanged.
BS	LEFT ARROW BS ^H ^]	Deletes the last character typed.
INSERT	insert ^R	Enters/exits insert mode.
NEWLINE	home ^K	Makes the current line the new template.

* Japanese. ^/in all other versions.

Example:

If you type the following command

DIR PROG.COM

MSX-DOS displays information about the file PROG.COM on your screen. The command line is also saved in the template. To repeat the command, just press two keys: <COPYALL> and <RETURN>.

The repeated command is displayed on the screen as you type, as shown below:

<COPYALL>DIR PROG.COM <RETURN>

Notice that pressing the <COPYALL> key causes the contents of the template to be copied to the command line; pressing <RETURN> causes the command line to be sent to the command processor for execution.

If you want to display information about a file named PROG.ASM, you can use the contents of the template and type:

<COPYUP>C

Typing <COPYUP>C copies all characters from the template to the command line, up to but not including "C". MSX-DOS displays:

DIR PROG.__

Note that the underline is your cursor. Now type:

.ASM

The result is:

DIR PROG.ASM__

The command line "DIR PROG. ASM" is now in the template and ready to be sent to the command processor for execution. To do this, press <RETURN>

Now assume that you want to execute the following command:

TYPE PROG.ASM

To do this, type:

TYPE <INSERT> <COPYALL> <RETURN>

Notice that when you are typing, the characters are entered directly into the command line and overwrite corresponding characters in the template. This automatic replacement feature is turned off when you press the insert key. Thus, the characters "TYPE" replace the characters "DIR" in the template. To insert a space between "TYPE" and "PROG.ASM", you press <INSERT> and then the space bar. Finally, to copy the rest of the template to the command line, you press <COPYALL> and then <RETURN>. The command "TYPE PROG.ASM" will be processed by MSX-DOS, and the template becomes "TYPE PROG.ASM".

If you had misspelled "TYPE" as "BYTE", a command error would have occurred. Still, instead of throwing away the whole command, you could save the misspelled line before you press <RETURN> by creating a new template with the <NEWLINE> key:

BYTE PROG.ASM <NEWLINE>

You could then edit this erroneous command by typing:

T <COPY1> P <COPYALL>

The <COPY1> key copies a single character from the template to the command line. The resulting command line is then the command that you want:

TYPE PROG.ASM

As an alternative, you can use the same template containing BYTE PROG.ASM and then use the <SKIP1> and <INSERT> keys to achieve the same result:

<SKIP1> <SKIP1> <COPY1> <INSERT> YP <COPYALL>

To illustrate how the command line is affected as you type, examine the keys typed on the left; their effect on the command line is shown on the right:

<SKIP1>	—	Skips over 1st template character
<SKIP1>	—	Skips over 2nd template character
<COPY1>	T	Copies 3rd template character
<INSERT> YP	TYP	Inserts two characters
<COPYALL>	TYPE PROG.ASM	Copies rest of template

Notice that <SKIP1> does not affect the command line. It affects the template by deleting the first character. Similarly, <SKIPUP> deletes characters in the template, up to but not including a given character.

These special editing keys can add to your effectiveness at the keyboard. The next section describes control character functions that can also help when you are typing commands.

II-2.13.2 Control Character Functions

A control character function is a function that affects the command line. You have already learned about <CONTROL-C> and <CONTROL-S>. Other control character functions are described below.

Remember that when you type a control character, such as <CONTROL-C>, you must hold down the control key and then press the "C" key.

Table of Control Character Functions

Control Character	Function
<CONTROL-N>	Cancels echoing of output to line printer.
<CONTROL-C>	Aborts current command.
<CONTROL-H>	Removes last character from command line, and erases character from terminal screen.
<CONTROL-J>	Inserts physical end-of-line, but does not empty command line. Use the <LINE FEED> key to extend the current logical line beyond the physical limits of one terminal screen.
<CONTROL-P>	Echoes terminal output to the line printer.
<CONTROL-S>	Suspends display of output to terminal screen. Press any key to resume.

II-2.14 Instructions for Users with Single-drive Systems

On a signel-drive system, you enter the commands as you would on a multi-drive system.

You should think of the single-drive system as having two drives (drive A and drive B). But instead of A and B representing two physical drives as on the multi-drive system, the A and B represent disks.

If you specify drive B when the "drive A disk" was last used, you are prompted to insert the disk for drive B. For example:

```
A>COPY COMMAND.COM B:
Insert diskette for drive B:
and strike a key when ready
 1 File(s) copied
A>_
```

If you specify drive A when the "drive B disk" was last used, you are prompted again to change disks. This time, MSX-DOS prompts you to insert the "drive A disk."

The same procedure is used if a command is executed from a batch file. MSX-DOS waits for you to insert the appropriate disk and to press any key before it continues. You will be prompted to do this.

NOTE

The letter displayed in the system prompt represents the default drive where MSX-DOS looks to find a file whose name is entered without a drive specifier. The letter in the system prompt does not represent the last disk used.

For example, assume that A is the default drive. If the last operation performed was DIR B:, MSX-DOS believes the "drive B disk" is still in the drive. However, the system prompt is still A>, because A is still the default drive. If you type DIR, MSX-DOS prompts you for the "drive A disk" because drive A is the default deive, and you did not specify another drive in the DIR command.

II-2.15 Disk Errors

If a disk error occurs at any time during a command or program, MSX-DOS retries the operation three times. If the operation cannot be completed successfully, MSX-DOS returns an error message in the following format:

```
<yyy> error <I/O action> drive X
Abort, Retry, Ignore?_
```

In this message, <yyy> may be one of the following:

```
Write protect
Not ready
Disk
```

The <I/O-action> may be either of the following:

```
reading
writing
```

The drive <x> indicates the drive in which the error has occurred.

MSX-DOS waits for you to enter one of the following responses:

A Abort. Terminate the program requesting the disk read or write.

I Ignore. Ignore the bad sector and pretend the error did not occur.

R Retry. Repeat the operation. This response is to be used when the operator has corrected the error.

Usually, you will want to attempt recovery by entering responses in this order:

R (to try again)
A (to terminate program and try a new disk)

One other error message might be related to faulty disk read or write:

```
Bad FAT
```

This message means that the copy in memory of one of the allocation tables has pointers to nonexistent blocks. Possibly the disk was incorrectly formatted or not formatted before use. If this error persists, the disk is currently unusable and must be formatted prior to use.

II-3. MSX-DOS Command Guide

NOTE

Users of single-drive systems should refer to 3.1.14 for the additional procedures required when executing many of the following commands.

The following MSX-DOS commands are described here. Note that synonyms for commands are enclosed in parentheses.

BASIC	Goto MSX-BASIC
COPY	Copies file(s) specified
DATE	Displays and sets date
DEL	Deletes file(s) specified (ERASE)
DIR	Lists requested directory entries
FORMAT	Formats a disk to receive MSX-DOS file
MODE	Sets display screen mode
PAUSE	Pauses for input in a batch file
REM	Displays a comment in a batch file
REN	Renames first file as second file (RENAME)
TIME	Displays and sets time
TYPE	Displays the contents of file specified

BASIC

SYNTAX : BASIC [<filespec>]

PURPOSE : Boots MSX-BASIC

COMMENTS: This command boots the MSX Disk BASIC from the MSX-DOS.

If a BASIC program file is designated by the filespec , the program is automatically loaded and run after BASIC starts.

This command changes the slot to make the BASIC ROM effective. So the memory map is different between the MSX-DOS and MSX-Disk-BASIC.

Use "CALL SYSTEM" (or-SYSTEM) statement to return to the MSX-DOS from the BASIC.

COPY

SYNTAX : COPY <filespec> [<filespec>]

PURPOSE : Copies one or more files to another disk. If you prefer, you can give the copies different names.
This command can also copy files on the same disk.

COMMENTS: If the second filespec option is not given, the copy will be on the default drive and will have the same name as the original file (first filespec option). If the first filespec is on the default drive and the second filespec is not specified, the COPY will be aborted. (Copying files to themselves is not allowed.)
MSX-DOS will return the error message:

File cannot be copied onto itself
0 files copied

The second option may take three forms:

1. If the second option is a drive designation (d:) only, the original file is copied with the original filename to the designated drive.
2. If the second option is a filename only, the original file is copied to a file on the default drive with the filename specified.
3. If the second option is a full filespec, the original file is copied to a file on the default drive with the filename specified.

The COPY command also allows file concatenation (joining) while copying. Concatenation is accomplished by simply listing any number of files as options to COPY, separated by "+"

For example,

COPY A.XYZ + B.COM + B:C.TXT BIGFILE.CRP

This command concatenates files named A.XYZ, B.COM, and B:C.TXT and places them in the file on the default drive called BIGFILE. CRP.

To combine several files using wild cards into one file, you could type:

```
COPY *.LST COMBIN.PRN
```

This command would take all files with a filename extension of .LST and combine them into a file named COMBIN.PRN.

In the following example, for each file found matching *.LST, that file is combined with the corresponding .REF file. The result is a file with the same filename but with the extension .PRN. Thus, FILE1.LST will be combined with FILE1.REF to form FILE1.PRN; then XYZ.LST with XYZ.REF to form XYZ.PRN; and so on.

```
COPY *.LST + *.PRN
```

The following COPY command combines all files matching *.LST, then all files matching *.REF, into one file named COMBIN.PRN:

```
COPY *.LST + *.REF COMBIN.PRN
```

Do not enter a concatenation COPY command where one of the source filenames has the same extension as the destination. For example, the following command is an error if ALL.LST already exists:

```
COPY *.LST ALL.LST
```

The error would not be detected, however, until ALL.LST is appended. At this point it could have already been destroyed.

COPY compares the filename of the input file with the filename of the destination. If they are the same, that one input file is skipped, and the error message "Content of destination lost before copy" is printed. Further concatenation proceeds normally. This allows "summing" files, as in this example:

```
COPY ALL.LST + .LST
```

This command appends all *.LST files, except ALL.LST itself, to ALL.LST. This command will not produce an error message and is the correct way to append files using the COPY command.

Because ASCII files are usually concatenated, this command interprets a CTRL+Z (1AH) as a end of file mark in a file. So there is a need of a "/B" switch to use a physical end of file (length of file displayed by the DIR command), when binary files shall be concatenated.

```
COPY/B A.COM+B. COM
```

In this example, the B.COM is appended after the A.COM, and the destination file name is still A.COM.

Any files can be concatenated by using "/B" switch for binary file and "/A" for ASCII file. A switch is effective for the switched file and the after until another switch appears.

Whether a CTRL+Z is appended at the end of the destination file or not is decided by a switch of the destination file. There is no CTRL+Z in the source file which is read in effect of "/A". Only one CTRL+Z is written when a file is written in effect of "/A". Therefore more CTRL+Z are appended as follows.

```
COPY A.ASM/B B.ASM/A
```

In this example, "/B" avoids removing CTRL+Z and "/A" appends a CTRL+Z.

When there is no concatenation, "/A" and "/B" switchs are valid, and the default file type is binary. "/A" switch terminates the copy at the first CTRL+Z.

COMO CREAR DESDE CONSOLA EL BCLOAD.

```
COPY CON BCLOAD
+4000 ←
A: ←
(CONTROL-Z) SALVA
```

COMO COPIAR UN DISCO

```
COPY *.*,B:
```

```
CON = PANTALLA
LST = PRINTER
```

DATE

SYNTAX : DATE [<yy>-<mm>-<dd>]

PURPOSE : Enter or change the date known to the system. This date will be recorded in the directory for any files you create or alter.

You can change the date from your terminal or from a batch file. (MSX-DOS does not display a prompt for the date if you use an AUTOEXEC.BAT file, so you may want to include a DATE command in that file.)

COMMENTS: If you type DATE, DATE will respond with the message:

```
Current date is <day>-<yy>-<mm>-<dd>
Enter new date:_
```

Press <RETURN> if you do not want to change the date shown.

You can also type a particular date after the DATE command, as in:

```
DATE 81-3-9
```

In this case, you do not have to answer the "Enter new date:" prompt.

The new date must be entered using numerals only; letters are not permitted. The allowed options are:

```
<mm> = 1-12
<dd> = 1-31
<yy> = 0-79, 80-99 or 1980-2099
```

The date, month, and year entries may be separated by hyphens (-), slashes (/) or periods (.). MSX-DOS is programmed to change months and years correctly, whether the month has 31, 30, 29, or 28 days. MSX-DOS handles leap years, too.

<yy> is a two-digit number from 80-99 (the 19 is assumed), or a two-digit number from 00-79 (the 20 is assumed), or a four-digit number from 1980-2099 (representing year.)

If the options or separators are not valid, DATE displays the message:

```
Invalid date
Enter new date:_
```

DATE then waits for you to enter a valid date.

DEL

SYNONYM : DELETE
ERASE

SYNTAX : DEL [filespec]

PURPOSE : Deletes all files with the designated filespec.

COMMENTS: If the filespec is *.*, the prompt "Are you sure?" appears. If a "Y" or "y" or <RETURN> is typed as a response, then all files are deleted as requested. You can also type ERASE for the DELETE command.

DIR

SYNTAX : DIR [filespec] [/P] [/W]

PURPOSE : Lists the files in a directory.

COMMENTS: If you just type DIR, all directory entries on the default drive are listed. If only the drive specification is given (DIR d:), all entries on the disk in the specified drive are listed. If only a filename is entered with no extension (DIR filename), then all files with the designated filename on the disk in the default drive are listed. If you designate a file specification (for example, DIR d:filename.ext), all files with the filename specified on the disk in the drive specified are listed. In all cases, files are listed with their size in bytes and with the time and date of their last modification.

The wild card characters ? and * (question mark and asterisk) may be used in the filename option.

Note that for your convenience the following DIR commands are equivalent:

COMMAND	EQUIVALENT
DIR	DIR *.*
DIR FILENAME	DIR FILENAME. *
DIR .EXT	DIR *.EXT
DIR .	DIR *.

Two switches may be specified with DIR. The /P switch selects Page Mode. With /P, display of the directory pauses after the screen is filled.
To resume display of output, press any key.

The /W switch selects Wide Display. With /W, only filenames are displayed, without other file information. Files are displayed as much as possible per line.

PARA IMPRIMIR
<CONTROL> P
FIN DE IMPRESION <CONTROL> N

FORMAT

SYNTAX : FORMAT

PURPOSE : Formats the disk in the specified drive to accept MSX-DOS files.

COMMENTS: This command initializes the directory and file allocation tables. A new disk must be formatted before use. If a used disk is formatted, all files in the disk are destroyed.

MSX-DOS issues the following message:

Drive name? (A,B) _

Select a drive name carefully. After you enter the drive name, the following message is displayed.

Strike a key when ready_

After you insert the new disk in the drive and press any key on the keyboard.

When the formatting finish, MSX-DOS will issue a following message.

Format complete

NOTE

The format procedure may be different with this description. For example, you can choose disk format from single side or double side with some disk driver. See your disk driver's manual.

MODE

SYNTAX : MODE <width>

PURPOSE : Sets the width of the display.

COMMENTS: <width> is the maximum number of characters per line on display.

<width> must be between 1 and 40. If it is 32 or less, screen mode 1 is selected, else mode 0 is selected.

The default screen mode and width of international MSX versions are as follows.

<table>
<tr><th>Version</th><th>Default screen mode</th><th>Default screen width</th></tr>
<tr><td>Korea</td><td rowspan="2">1</td><td rowspan="2">29</td></tr>
<tr><td>Japan</td></tr>
<tr><td>USA</td><td rowspan="5">0</td><td>39</td></tr>
<tr><td>UK</td><td rowspan="4">37</td></tr>
<tr><td>DIN</td></tr>
<tr><td>French</td></tr>
<tr><td>INT</td></tr>
</table>

PAUSE

SYNTAX : PAUSE [comment]

PURPOSE : Suspends execution of the batch file.

COMMENTS: During the execution of a batch file, you may need to change disks or perform some other action. PAUSE suspends execution until you press any key, except <CONTROL-C>.

When the command processor encounters PAUSE, it prints:

Strike a key when ready . . .

If you press <CONTROL-C>, another prompt will be displayed:

Terminate batch file (Y/N)?

If you type "Y" in response to this prompt, execution of the remainder of the batch command file will be aborted and control will be returned to the operating system command level. Therefore, PAUSE can be used to break a batch file into pieces, allowing you to end the batch command file at an intermediate point.

The comment is optional and may be entered on the same line as PAUSE. You may also want to prompt the user of the batch file with some meaningful message when the batch file pauses. For example, you may want to change disks in one of the drives. An optional prompt message may be given in such cases. The comment prompt will be displayed before the "Strike a key" message.

REM

SYNTAX : REM [comment]

PURPOSE : Displays remarks which are on the same line as the REM command in a batch file during execution of that batch file.

COMMENTS: The only separators allowed in the comment are the space, tab, and comma.

REN

SYNONYM : RENAME

SYNTAX : REN <filespec> <filename>

PURPOSE : Changes the name of the first option (filespec) to the second option (filename).

COMMENTS: The first option (filespec) must be given a drive designation if the disk resides in a drive other than the default drive. Any drive designation for the second option (filename) is ignored. The file will remain on the disk where it currently resides.

The wild card characters may be used in either option. All files matching the first filespec are renamed. If wild card characters appear in the second filename, corresponding character positions will not be changed.

For example, the following command changes the names of all files with the .LST extension to similar names with the .PRN extension:

```
REN *.LST*.PRN
```

In the next example, REN renames the file ABODE on drive B to ADOBE:

```
REN B:ABODE ?D ?B ?
```

The file remains on drive B.

An attempt to rename a filespec to a name already present in the directory will result in the error message "Rename error"

TIME

SYNTAX : TIME [<hh> [: <mm> [: <ss>]]]

PURPOSE : Displays and sets the time.

COMMENTS: If the TIME command is entered without any arguments, the following message is displayed:

```
Current time is  <hh> : <mm> : <ss> . <cc>

Enter new time:_
```

Press the <RETURN> key if you do not want to change the time shown. A new time may be given as an option to the TIME command as in:

```
TIME 8:20
```

The new time must be entered using numerals only; letters are not allowed. The allowed options are:

<hh> = 00-24
<mm> = 00-59
<ss> = 00-59

The hour and minute entries must be separated by colons. You do not have to type the <ss> (seconds) or <cc> (hundredths of seconds) options.

MSX-DOS uses the time entered as the new time if the options and separators are valid. If the options or separators are not valid, MSX-DOS displays the message:

```
Invalid time
Enter new time:_
```

MSX-DOS then waits for you to type a valid time.

NOTE

If your computer does not have a clock, this command is nonsense.

TYPE

SYNTAX : TYPE <filespec>

PURPOSE : Displays the contents of the file on the console screen.

COMMENTS: Use this command to examine a file without modifying it. (Use DIR to find the name of a file.) The only formatting performed by TYPE is that tabs are expanded to spaces consistent with tab stops every eighth column. Note that a display of binary files causes control characters (such as CONTROL-Z) to be sent to your computer, including bells, form feeds, and escape sequences.

PART III.
MSX DISK-BASIC

III-1. MSX Disk BASIC Reference Guide

Microsoft (TM) BASIC is the most extensive implementation of BASIC available for microprocessors. Microsoft BASIC meets the ANSI qualifications for BASIC, as set forth in document BSRX3.60-1978. Each release of Microsoft BASIC is compatible with previous versions.

MSX (TM) disk BASIC is a release of Microsoft BASIC for the MSX computer and its flexible disk system.

III-1.1 Commands and Statements

BLOAD

SYNTAX : BLOAD "< filespec >" {[,R] | [,S]} [,offset]

PURPOSE : Loads a machine language program or an array from disk or cassette tape into memory.

COMMENTS : The file name can be omitted only for the file in the cassette tape, not for the disk.

If no < offset > is specified, the program is loaded from the address designated by the BSAVE command. If an < offset > is specified, the program is loaded from the address added < offset > to the saved address. Programs to be loaded with the offset must be relocatable.

The R option automatically runs the program after it has been loaded.

The S option loads the screen image saved by the "BSAVE ,S" statement to video RAM.

If no drive name is specified, the program in the current drive is loaded.

See also "BSAVE,".

EXAMPLE : BLOAD "MAX2"

Loads file "MAX2" into memory.

BSAVE

SYNTAX : BSAVE "< filespec >", < start address > , < end address >
[, < execute address >] I [,S]

PURPOSE : Saves the machine language program currently in memory on disk or cassette tape.

COMMENTS: The program from < start address > to < end address > in memory is saved on disk or cassette tape.

If no drive name is specified, the program is saved on the current drive.

start address > defines the default execution address.

Ths S option saves the content of video RAM to the file.

See also "BLOAD,".

EXAMPLE : BSAVE "TIMER", &HC000, &HCFFF

Saves the program currently in memory from &HC000 to &HCFFF on current drive under filename "TIMER".

CLOSE

SYNTAX : CLOSE [[#] < file number > [, [#] < file number... >]]

PURPOSE : Concludes I/O to a disk file.

COMMENTS: < file number > is the number under which the file was OPENed. A CLOSE with no arguments closes all open files.

The association between a particular file and file number terminates upon execution of a CLOSE statement. The file may then be reOPENed using the same or a different file number; likewise, that file number may now be reused to OPEN any file.

A CLOSE for a sequential output file writes the final buffer of output.

The END, CLEAR statements and the NEW command always CLOSE all disk files automatically.
(STOP does not close disk files.)

EXAMPLE : CLOSE #1

COPY

SYNTAX : COPY "<file spec>" TO "<file spec>"

PURPOSE : Copies one or more files to another disk. If you prefer, you can give the copies different names.
This command can also copy files on the same disk.

COMMENTS: The second option may take three forms:

1. If the second option is a drive designation (d:) only, the original file is copied with the original filename to the designated drive.
2. If the second option is a filename only, the original file is copied to a file on the default drive with the filename specified.
3. If the second option is a full filespec, the original file is copied to a file on the default drive with the filename specified.

On a single-drive system, you enter the commands as you would on a multi-drive system.

If you specify drive B when the "drive A disk" was last used, you are prompted to insert the disk for drive B. For example:

```
COPY "A:TEST.ASC" TO "B:"
```

After the file is loaded from "drive A disk" to memory, you are prompted as follows.

```
Insert diskette for drive B:
and strike a key when ready
```

You remove "A disk" and insert "B disk". Then strike any key (except CONTROL-STOP). If the file is small, copy is completed.

But, if the file is big, you must exchange two disks following the prompted instructions until copy is completed. Because parts of the file are loaded and saved one after another.

If you specify drive A when the "drive B disk" was last used, you are prompted again to change disks. This time, BASIC prompts you to insert the "drive A disk". See also section 3.1.14.

DSKO

SYNTAX : DSKO <drive number>, <logical sector number>

COMMENTS: Writes to the specified sector from memory pointed to by the content of (0F351H, 0F352H).

<drive number> is 0 for default drive, 1 for drive A, 2 for drive B, and so on.

<logical sector number> is a 0 based number. No check for the valid sector number is made.

NOTE : This memory area is destroyed when any disk statements (ex. FILES, OPEN, CLOSE, PRINT#, etc.) are executed.

FIELD

SYNTAX : FIELD [#] <file number>, <field width>

AS <string variable>...

PURPOSE : Allocates space for variables in a random file buffer.

COMMENTS: Before a GET statement or PUT statement can be executed, a FIELD statement must be executed to format the random file buffer.

<file number> is the number under which the file was OPENed. <field width> is the number of characters to be allocated to <string variable>.
For example,

```
FIELD 1,20 AS N$,10 AS ID$,40 AS ADD$
```

allocates the first 20 positions (bytes) in the random file buffer to the string variable N$, the next 10 positions to ID$, and the next 40 positions to ADD$. FIELD does NOT place any data in the random file buffer. (See "LSET/RSET,", and "GET,".)

The total number of bytes allocated in a FIELD statement must not exceed the record length that was specified when the file was OPENed. Otherwise, a "Field overflow" error occurs. (The default record length is 256 bytes.)

Any number of FIELD statements may be executed for the same file. All FIELD statements that have been executed will remain in effect at the same time.

NOTE : Do not use a FIELDed variable name in an INPUT or LET statement. Once a variable name is FIELDed, it points to the correct place in the random file buffer. If a subsequent INPUT or LET statement with that variable name is executed, the variable's pointer is moved to string space.

EXAMPLE 1 :
```
10 OPEN "A:PHONELST" AS #1 LEN=35
15 FIELD #1,2 AS RECNBR$,33 AS DUMMY$
20 FIELD #1,25 AS NAMES,10 AS PHONENBR$
25 GET #1
30 TOTAL=CVI (RECNBR)$
35 FOR I=2 TO TOTAL
40 GET #1, I
45 PRINT NAMES, PHONENBR$
50 NEXT I
```

Illustrates a multiple defined FIELD statement. In statement 15, the 35 byte field is defined for the first record to keep track of the number of records in the file. In the next loop of statements (35-50), statement 20 defines the field for individual names and phone numbers.

EXAMPLE 2:
```
10 FOR LOOP%=0 TO 7
20 FIELD #1, (LOOP%*16) AS OFFSETS, 16 AS A$ (LOOP%)
30 NEXT LOOP%
```

Shows the construction of a FIELD statement using an array of elements of equal size. The result is equivalent to the single declaration:

```
FIELD #1,16 AS A$(0), 16 AS A$(1),...,16 AS A$(6), 16 AS A$(7)
```

EXAMPLE 3:
```
10 DIM SIZE% (NUMB%) : REM ARRAY OF FIELD SIZES
20 FOR LOOP%=0 TO NUMB% : READ SIZE% (LOOP%) : NEXT
                                              LOOP%
30 DATA 9, 10, 12, 21, 41
.
.
.
120 DIM A$ (NUMB%) : REM ARRAY OF FIELDED VARIABLES
130 OFFSET%=0
140 FOR LOOP%=0 TO NUMB%
150 FIELD #1, OFFSET% AS OFFSET$, SIZE% (LOOP%)
    AS A$ (LOOP%)
160 OFFSET%=OFFSET%+SIZE% (LOOP%)
170 NEXT LOOP%
```

Creates a field in the same manner as Example 2. However, the element size varies with each element. The equivalent declaration is:

```
FIELD #1, SIZE%(0) AS A$(0), SIZE%(1) AS A$(1),...
SIZE% (NUMB%) AS A$ (NUMB%)
```

UN DIR ELEMENTAL

FILES and LFILES

SYNTAX : FILES ["< file spec >"]
LFILES ["< file spec >"]

PURPOSE : Displays or prints file names of disk files.

COMMENTS: The file names designated by the < file spec > are displayed. If the designated file does not exists, "File not found" error is occurs.

If no < file spec > is specified, all file names in the current drive are displayed.

There can be question mark (?) in the file name to substitute for a character in the file name or extension. And, there can be asterisk (*) to substitute for any file name or extension.

If the drive name is designated, the file names in that drive is displayed, else in current drive.

The LFILES command outputs file names not to display but to printer

EXAMPLE : FILES "B:.BAS"

FORMAT

SYNTAX : CALL FORMAT
or
_FORMAT

PURPOSE : Initializes a disk.

COMMENTS: Menu is displayed as follows.

Drive name? (A,B) _

Select a drive name carefully. After you enter the drive name, the following message is displayed.

Strike a key when ready_

After you insert the new disk in the drive and press any key on the keyboard.

When the formatting is finished, BASIC will issue the following message.

Format complete

NOTE : If a used disk is formatted, all files in that disk is destroyed.

New disks must be formatted before use.

The format procedure may be different with this description. For example, you can choose disk format from single side or double side with some disk driver. See your disk driver's manual.

GET

SYNTAX : GET [#] < file number > [, < record number >]

PURPOSE : Reads a record from a random disk file into a random buffer.

COMMENTS: < file number > is the number under which the file was OPENed. If < record number > is omitted, the next record (after the last GET) is read into the buffer. The largest possible record number is 4,294,967,295.

EXAMPLE :
```
10 OPEN "SAMPLE.DAT" AS #1
20 FIELD #1, 2 AS A$, 10 AS B$
30 FOR I%=1 TO 10
40 GET #1, I%
50 PRINT CVI(A$) ; B$
60 NEXT
70 CLOSE #1
80 END
```

NOTE : After an execution of a GET statement, INPUT# and LINE INPUT# may be executed to read characters from the random file buffer.

INPUT#

SYNTAX : INPUT# < file number >, < variable list >

PURPOSE : Reads data items from a sequential disk file and assigns them to program variables.

COMMENTS: < file number > is the number used when the file was OPENed for input. < variable list > contains the variable names that will be assigned to the items in the file. (The variable type must match the type specified by the variable name.)
With INPUT#, no question mark is printed, as with INPUT.

The data items in the file should appear just as they would if data were being typed in response to an INPUT statement. With numeric values, leading spaces, carriage returns, and line feeds are ignored. The first character encountered that is not a space, carriage return, or line feed is assumed to be the start of a number. The number terminates on a space, carriage return, line feed, or comma.

If MSX BASIC is scanning the sequential data file for a string item, leading spaces, carriage returns, and line feeds are also ignored. The first character encountered that is not a space, carriage return, or line feed is assumed to be the start of a string item. If this first character is a quotation mark ("), the string item will consist of all characters read between the first quotation mark and the second. Thus, a quoted string may not contain a quotation mark as a character. If the first character of the string is not a quotation mark, the string is an unquoted string, and will terminate on a comma, a carriage return, or a line feed (or after 255 characters have been read). If end-of-file is reached when a numeric or string item is being INPUT, the item is terminated.

EXAMPLE :
```
10 OPEN "SAMPLE2.DAT" FOR INPUT AS #1
20 INPUT #1, A$
30 PRINT A$
40 IF EOF(1)=0 THEN 20
50 CLOSE #1
60 END
```

KILL

SYNTAX : KILL " < file spec > "

PURPOSE : Deletes a file from disk.

COMMENTS: If a KILL statement is given for a file that is currently OPEN, a "File already open" error occurs.

KILL is used for all types of disk files: program files, random data files, and sequential data files.

EXAMPLE : 200 KILL "DATA1.DAT"

LINE INPUT#

SYNTAX : LINE INPUT# < file number >, < string variable >

PURPOSE : Reads an entire line (up to 254 characters), without delimiters, from a sequential disk data file to a string variable.

COMMENTS: < file number > is the number under which the file was OPENed. < string variable > is the variable name to which the line will be assigned. LINE INPUT# reads all characters in the sequential file up to a carriage return. It then skips over the carriage return/line feed sequence. The next LINE INPUT# reads all characters up to the next carriage return. (If a line feed/carriage return sequence is encountered, it is understood as a string ending with a line feed character.)

LINE INPUT# is especially useful if each line of a data file has been broken into fields, or if an MSX BASIC program saved in ASCII format is being read as data by another program. (See "SAVE,".)

EXAMPLE : 10 OPEN "LIST" FOR OUTPUT AS #1
20 LINE INPUT "CUSTOMER INFORMATION?" ; C$
30 PRINT #1, C$
40 CLOSE 1
50 OPEN "LIST" FOR INPUT AS #1
60 LINE INPUT #1, C$
70 PRINT C$
80 CLOSE 1
RUN
CUSTOMER INFORMATION? LINDA JONES
234,4 MEMPHIS LINDA JONES 234,4 MEMPHIS
Ok

LOAD

SYNTAX : LOAD < filename > [,R]

PURPOSE : Loads a file from disk into memory.

COMMENTS: < filename > is the name that was used when the file was SAVEd.

The R option automatically runs the program after it has been loaded.

LOAD closes all open files and deletes all variables and program lines currently residing in memory before it loads the designated program. However, if the R option is used with LOAD, the program is RUN after it is LOADed, and all open data files are kept open. Thus, LOAD with the R option may be used to chain several programs (or segments of the same program). Information may be passed between the programs using their disk data files.

Until the designated file is found and start being loaded, the program in memory is kept.

EXAMPLE : LOAD "STRTRK", R

LOAD "B:MYPROG"

LSET and RSET

SYNTAX : LSET < string variable > = < string expression >
RSET < string variable > = < string expression >

PURPOSE : Moves data from memory to a random file buffer (in preparation for a PUT statement).

COMMENTS: If < string expression > requires fewer bytes than were FIELDed to < string variable >, LSET left-justifies the string in the field, and RSET right-justifies the string. (Spaces are used to pad the extra positions.) If the string is too long for the field, characters are dropped from the right. Numeric values must be converted to strings before they are LSET or RSET. (See "MKI$, MKS$, MKD$,".)

EXAMPLE : 150 LSET A$=MKS$ (AMT)
160 LSET D$=DESC($)

NOTE : LSET or RSET may also be used with a nonfielded string variable to left-justify or right-justify a string in a given field. For example, the program lines

```
110 A$=SPACE$(20)
120 RSET A$=N$
```

right-justifies the string N$ in a 20-character field. This can be very handy for formatting printed output.

MAXFILES

SYNTAX : MAXFILES= < expression >

PURPOSE : Specifies the maximum number of files opened at a time.

COMMENTS: < expression > can be in the range of 0 to 15. When 'MAXFILES=0' is executed, only SAVE and LOAD can be performed.

MERGE

SYNTAX : MERGE < filename >

PURPOSE : Merges a specified disk file into the program currently in memory.

COMMENTS: < filename > is the name used when the file was SAVEd. The file must be SAVEd in ASCII format. (If not, a "Bad file mode" error occurs.)

If any lines in the disk file have the same line numbers as lines in the program in memory, the lines from the file on disk will replace the corresponding lines in memory. (MERGEing may be thought of as "inserting" the program lines on disk into the program in memory.)

MSX BASIC always returns to command level after executing a MERGE command.

EXAMPLE : MERGE "NUMBRS"

NAME

SYNTAX : NAME < old filespec > AS < new filename >

PURPOSE : Changes the name of a disk file.

COMMENTS: < old filespec > must exist and < new filename > must not exist; otherwise, an error will result. After a NAME command, the file exists on the same disk, in the same area of disk space, with the new name.

If no drive name is specified, the current drive is selected.

EXAMPLE : NAME "ACCTS" AS "LEDGER"

In this example, the file that was formerly named ACCTS will now be named LEDGER.

OPEN

SYNTAX : OPEN "< filespec >" [FOR < mode >] AS [#] < file number > [LEN= < reclen >]

PURPOSE : Allows I/O to a disk file.

COMMENTS: A disk file must be opened before any disk I/O operation can be performed on that file. OPEN allocates a buffer for I/O to the file and determines the mode of access that will be used with the buffer.

< mode > is one of the following:

FOR OUTPUT	Specifies sequential output mode.
FOR INPUT	Specifies sequential input mode.
FOR APPEND	Specifies sequential append mode after end of an existent file.
default	Specifies random input/output mode.

< file number > is an integer expression whose value is between one and the maximum number of files specified in a MAXFILES statement. The number is then associated with the file as long as it is OPEN and is used to refer to other disk I/O statements to the file.

< filename > is a string expression containing a name that conforms to your operating system's rules for disk filenames.

< reclen > is an integer expression which, if included, sets the record length for random files. The default record length is 256 bytes. The largest possible record length is 256. The smallest is 1.

NOTE : If sequential input or append mode is used for non-existent file, "File not found" error occurs. If sequential output mode is used for existent file, the old file is deleted.

A file can be OPENed for sequential input or random access on more than one file number at a time. A file may be OPENed for output, however, on only one file number at a time.

EXAMPLE : 10 OPEN "INVEN" FOR INPUT AS #1

PRINT# and PRINT# USING

SYNTAX : PRINT# < file number >, [USING < string exp > ;] < list of expressions >

PURPOSE : Writes data to a sequential disk file.

COMMENTS: < file number > is the number used when the file was OPENed for output. < string exp > consists of formatting characters as described in "PRINT USING." The expressions in < list of expressions > are the numeric and/or string expressions that will be written to the file.

PRINT# does not compress data on the disk. An image of the data is written to the disk, just as it would be displayed on the terminal screen with a PRINT statement. For this reason, care should be taken to delimit the data on the disk, so that it will be inputted correctly from the disk.

In the list of expressions, numeric expressions should be delimited by semicolons. For example:

```
PRINT#1,A;B;C;X;Y;Z
```

(If commas are used as delimiters, the extra blanks that are inserted between print fields will also be written to the disk.)

String expressions must be separated by semicolons in the list. To format the string expressions correctly on the disk, use explicit delimiters in the list of expressions.

For example, let A$="CAMERA" and B$="93604-1".
The statement

```
PRINT#1,A$;B$
```

would write CAMERA93604-1 to the disk. Because there are no delimiters, this could not be input as two separate strings. To correct the problem, insert explicit delimiters into the PRINT# statement as follows:

```
PRINT#1,A$;",";B$
```

The image written to disk is

```
CAMERA,93604-1
```

which can be read back into two string variables. If the strings themselves contain commas, semicolons, significant leading blanks, carriage returns, or line feeds, write them to disk surrounded by explicit quotation marks, CHR$(34).

For example, let A$="CAMERA, AUTOMATIC" and B$=" 93604-1". The statement

PRINT#1,A$;B$

would write the following image to disk:

CAMERA, AUTOMATIC 93604-1

And the statement

INPUT#1,A$,B$

would input "CAMERA" to A$ and "AUTOMATIC 93604-1" to B$. To separate these strings properly on the disk, write double quotation marks to the disk image using CHR$(34). The statement

PRINT#1,CHR$(34) ; A$; CHR$(34) ; CHR$(34) ; B$;CHR$(34)

writes the following image to disk:

"CAMERA, AUTOMATIC" "93604-1"

And the statement

INPUT#1,A$,B$

would input "CAMERA, AUTOMATIC" to A$ and "93604-1" to B$.

The PRINT# statement may also be used with the USING option to control the format of the disk file. For example:

PRINT#1, USING "YY####.##," ;J;K;L

(Japanase. Refer to 5.4 for other versions.)

PUT

SYNTAX : PUT [#] < file number > [, < record number >]

PURPOSE : Writes a record from a random buffer to a random disk file.

COMMENTS: < file number > is the number under which the file was OPENed. If < record number > is omitted, the record will assume the next available record number (after the last PUT). The largest possible record number is 4,294,967,295. The smallest record number is 1.

EXAMPLE :
```
10 OPEN "SAMPLE.DAT" AS #1
20 FIELD #1, 2 AS A$, 10 AS B$
30 FOR I%=1 TO 10
40 INPUT N%, S$
50 LSET A$=MKI$(N%)
60 LSET B$=S$
70 PUT #1, I%
80 NEXT
90 CLOSE #1
100 END
```

NOTE : LSET or RSET statement must be used to put characters in the random file buffer before executing a PUT statement.

Any attempt to read or write past the end of the buffer causes a "Field overflow" error.

RUN

SYNTAX : RUN < filename > [,R]

PURPOSE : Loads a file from disk into memory and runs it.

COMMENTS: < filename > is the name used when the file was SAVEd.

RUN closes all open files and deletes the current contents of memory before loading the designated program. However, with the "R" option, all data files remain OPEN.

EXAMPLE : RUN "NEWFIL", R

SAVE

SYNTAX : SAVE <filespec> [,A]

PURPOSE : Saves a program file on disk.

COMMENTS: <filespec> is a quoted string that conforms to MSX-DOS's requirements for filenames. If <filespec> already exists, the file will be written over.

Use the A option to save the file in ASCII format. Otherwise, MSX BASIC saves the file in a compressed binary format. ASCII format takes more space on the disk, but some disk access requires that files be in ASCII format. For instance, the MERGE command requires an ASCII format file, and some operating system commands such as LIST may require an ASCII format file.

NOTE : "CSAVE" and "SAVE" are used for binary and ASCII save cassete tape file. But "SAVE" and "SAVE...,A" are used for that cases of disk file.

EXAMPLE : SAVE "COM2", A

SYSTEM

SYNTAX : CALL SYSTEM
or
_SYSTEM

PURPOSE : Exits from disk BASIC and returns to MSX-DOS.

COMMENTS: This command is valid only when BASIC has been booted from MSX-DOS.

By this command all files are closed and the program and the data in memory are destroyed.

CVI, CVS, CVD

SYNTAX : CVI (2-byte string ≥)
CVS (≤ 4-byte string >)
CVD (< 8-byte string >)

PURPOSE : To convert string values to numeric values. Numeric values that are read in from a random disk file must be converted from strings back into numbers. CVI converts a 2-byte string to an integer. CVS converts a 4-byte string to a single precision number. CVD converts an 8-byte string to a double-precision number.

EXAMPLE :

```
.
.
.
70 FIELD #1,4 AS N$, 12 AS B$, ...
80 GET #1
90 Y=CVS (N$)
.
.
.
```

See also "MKI$, MKS$, MKD$,".

DSKF

SYNTAX : DSKF (< drive number >)

PURPOSE : To know free area size of specified disk by K byte.

The drive number corresponds to the drive name as follows.

0 default drive
1 drive A:
2 drive B:
and so on

EXAMPLE : PRINT DSKF (1)

DSKI$

SYNTAX : DSKI$ (< drive number >, < logical sector number >)

PURPOSE : To read the specified sector to memory pointed to by the content of (0F351H, 0F352H).

< drive number > is 0 for default drive, 1 for drive A, 2 for drive B, and so on.

≤ logical sector number > is a 0 based number. No check for the valid sector number is made.

NOTE : This memory area is destroyed when any disk statements (ex. FILES, OPEN, CLOSE, PRINT#, etc.) are executed.

EOF

SYNTAX : EOF (< file number >)

PURPOSE : To know if the end of a sequential file has been reached. Returns -1 (true) if so. Use EOF to test for end-of-file while INPUTting, to avoid "Input past end" errors.

The file specified by the file number must be opened as sequential input mode.

EXAMPLE :

```
10 OPEN "DATA" FOR INPUT AS #1
20 C=0
30 IF EOF (1) THEN 100
40 INPUT #1,M(C)
50 C=C+1:GOTO 30
```

INPUT$

SYNTAX : INPUT$ (X [, [#] Y])

PURPOSE : To read data from the terminal or from file number Y. Returns a string of X characters. If the terminal is used for input, no characters will be echoed. All control characters are passed through except Control-STOP, which is used to interrupt the execution of the INPUT$ function.

EXAMPLE :
```
5 'LIST THE CONTENTS OF A SEQUENTIAL FILE IN HEXADECIMAL
10 OPEN "DATA" FOR INPUT AS #1
20 IF EOF(1) THEN 50
30 PRINT HEX$ (ASC (INPUT$(1,#1)));
40 GOTO 20
50 PRINT
60 END
```

LOC

SYNTAX : LOC (< file number >)

where < file number > is the number under which the file was OPENed.

PURPOSE : With random disk files, LOC returns the record number just read or written from a GET or PUT statement. If the file was opened but no disk I/O has been performed yet, LOC returns a 0. With sequential files, LOC returns the number of records read from or written to the file since it was OPENed. When no record is read from the sequential input file since it was opened, LOC returns 1, because SYSTEM has read the first sector.

EXAMPLE : 200 IF LOC(1) > 50 THEN STOP

LOF

SYNTAX : LOF (< file number >)

PURPOSE : LOF returns the size of the specified file by byte.

EXAMPLE : IF NUM% > LOF (1) THEN PRINT "INVALID"

MKI$, MKS$, MKD$

SYNTAX : MKI$ (< integer expression >)
MKS$ (< single precision expression >)
MKD$ (< double precision expression >)

PURPOSE : To convert numeric values to string values. Any numeric value that is placed in a random file buffer with an LSET or RSET statement must be converted to a string. MKI$ converts an integer to a 2-byte string. MKS$ converts a single precision number to a 4-byte string. MKD$ converts a double precision number to an 8-byte string.

EXAMPLE :
```
90 AMT=(K+T)
100 FIELD #1,8 AS D$,20 AS N$
110 LSET D$=MKS$ (AMT)
120 LSET N$=A$
130 PUT #1
```

See also "CVI, CVS, CVD."

VARPTR

SYNTAX : VARPTR (# < file number >)

PURPOSE : VARPTR returns the address of the file control block assigned to < file number >

EXAMPLE : 100 X=USR (VARPTR (#1))

III-1.3 Error Codes and Error Messages

Code Number Disk Errors Message

50 Field overflow

A FIELD statement is attempting to allocate more bytes than were specified for the record length of a random file.

51 Internal error

An internal malfunction has occurred in MSX BASIC. Report to Microsoft the conditions under which the message appeared.

52 Bad file number

A statement or command references a file with a file number that is not OPEN or is out of the range of file numbers specified at initialization.

53 File not found

A LOAD, KILL, or OPEN statement references a file that does not exist on the current disk.

54 File already open

A sequential output mode OPEN statement is issued for a file that is already open; or a KILL statement is given for a file that is open.

55 Input past end

An INPUT statement is executed after all the data in the file has been INPUT, or for a null (empty) file. To avoid this error, use the EOF function to detect the end-of-file.

56 Bad file name

An illegal form is used for the filename with a LOAD, SAVE, KILL, or OPEN statement (e.g., a filename with too many characters).

57 Direct statement in file

A direct statement is encountered while LOADing an ASCII-format file. The LOAD is terminated.

58 Sequential I/O only

A GET or PUT statement is used on a sequential file.

59 File not open

An input or output statement is executed on a not opened file.

60 Bad allocation table

The disk is not initialized.

61 Bad file mode

An attempt is made to use PUT, GET, or LOF with a sequential file, to LOAD a random file, or to execute an OPEN statement with a file mode other than "FOR INPUT", "FOR OUTPUT", "FOR APPEND" or default (random).

62 Bad drive name

A invalid drive name is specified.

64 File still open

The file is not closed.

65 File already exists

The filename specified in a NAME statement is identical to a filename already in use on the disk.

66 Disk full

All disk storage space is in use.

67 Too many files

An attempt is made to create a new file (using SAVE or OPEN) when all 255 directory entries are full.

68 Disk write protected

A PUT or PRINT# statement is executed on a write protected disk.

69 Disk I/O error

An I/O error occurred on a disk I/O operation. It is a fatal error; i.e., the operating system cannot recover from the error.

70 Disk offline

There is no disk in the specified drive.

71 Rename across disk

A RENAME statement is executed, across one drive to another.